

Measuring Divergence in Dependency Trees to Improve Projection Algorithms

Ryan Georgi · Fei Xia · William D. Lewis

Received: date / Accepted: date

Abstract Obtaining syntactic parses is a crucial part of many NLP pipelines. However, most of the world's languages do not have large amounts of syntactically annotated corpora available for building parsers. Syntactic projection techniques attempt to address this issue by using parallel corpora consisting of resource-poor and resource-rich language pairs, taking advantage of a parser for the resource-rich language and word alignment between the languages to project the parses onto the data for the resource-poor language. These projection methods can suffer, however, when the two languages are divergent. In this paper, we investigate the possibility of using small, parallel, annotated corpora to automatically detect divergent structural patterns between two languages. These patterns can then be used to improve structural projection algorithms, allowing for better performing NLP tools for resource-poor languages, in particular those that may not have large amounts of annotated data necessary for traditional, fully-supervised methods. While this detection process is not exhaustive, we demonstrate that common patterns of divergence can be identified automatically without prior knowledge of a given language pair, and the patterns can be used to improve performance of projection algorithms.

Keywords Multilingualism · Translation Divergence · Syntactic Projection

R. Georgi
F. Xia
University of Washington Department of Linguistics
Box 354340 Seattle, WA 98195-4340
Tel.: +1 (206) 543-2046
Fax: +1 (206) 685-7978
E-mail: rgeorgi@uw.edu
E-mail: fxia@uw.edu

W. D. Lewis
Microsoft Research, Bldg 99
14820 NE 36th St, Redmond, WA 98052-6399
E-mail: wilewis@microsoft.com

1 Introduction

When it comes to resources for natural language processing, a small handful of languages account for the vast majority of available resources. Out of the resources listed by the LRE Map (Calzolari et al., 2012), English accounts for 30% of all recorded resources, and the ten most resourced languages for 62% of all resources. A broad variety of tools are available for these resource-rich languages, as the time and effort spent to annotate resources for these languages allows for state-of-the-art systems to be built utilizing supervised and semi-supervised methods.

The availability of such resources is the result of a large investment over many years on a per-language-basis. Because creating high-quality annotation is expensive and labor intensive, the vast majority of the world’s languages lack such resources and high-performance NLP tools. To address this issue, recent studies (Lewis and Xia, 2008; Benajiba and Zitouni, 2010; Georgi et al., 2012) have proposed to take advantage of bitext and resources for resource-rich languages; that is, use tools for resource-rich languages to process one side of bitext (the resource-rich language) and project the information to the other side of bitext (the resource-poor language) via word alignments.

While projecting annotation from one language to another is a promising method for adding annotation to languages using automated methods, it relies on the assumption that simple word alignments between languages are sufficient to represent analogous meanings and structures between the languages. For reasons we will discuss in the following sections, this assumption is useful, but often erroneous.

Finding out whether and when this assumption fails for a given language pair is not easy without knowledge about the two languages. It would be useful if, given a small set of seed data, a language pair could be analyzed to find where and in what ways the languages diverge, and use these detected patterns as corrective guidelines for performing projection upon other sentences of the language pair.

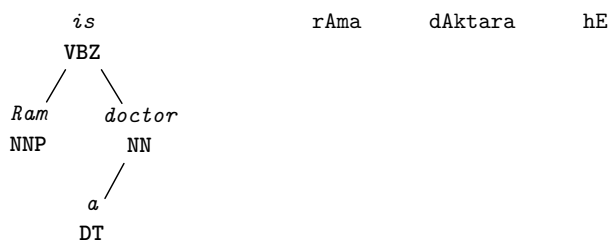
In this paper, we propose a method for analyzing a language pair and determining the degree and types of divergence between two languages. This systematic identification of divergence types could then lead to better informed syntactic projections, and subsequently can improve the tools built upon such data.

2 Background

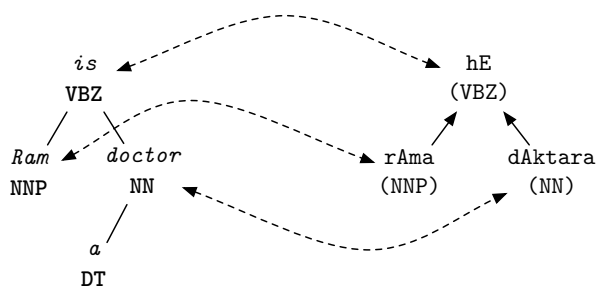
While there is a growing body of work on projection methods as a means to bootstrap resources from one language to another, there are not many studies on how to treat the issue of linguistic divergence between these languages. In this section, we provide a brief review of work on divergence and projection algorithms. We will also introduce interlinear glossed text (IGT), a common format for linguists to represent language examples.

rAma	dAktara	hE	Source
Ram	doctor	1.PRES	Gloss
"Ram	is	a doctor"	Translation

Fig. 1 An instance of Interlinear Glossed Text (IGT) for Hindi, with a gloss line and translation in English. The word alignment between the gloss and translation lines is not part of the IGT instance, but it can be produced by a statistical word aligner trained on bitext or an aligner that uses heuristic rules.



(a) Using the interlinear instance from Figure 1, the English text is parsed.



(b) Using the word alignments from Figure 1, the tree structure and POS tags for the English tree are “projected” onto the Hindi sentence.

Fig. 2 A simple example of syntactic projection as performed on the IGT instance in Figure 1.

2.1 Projection Methods

Projection algorithms have been the target of a fair amount of research in the last decade as attempts have been made to utilize statistical alignment methods to match words between languages with parallel data and “project” annotation between the two. Figure 1 shows an example bitext in the form of an interlinear glossed text (IGT), while Figure 2 shows how this bitext may be used to project a dependency tree from the English to the Hindi.

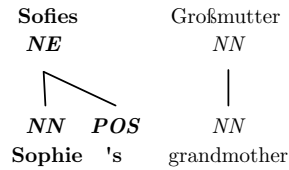


Fig. 3 Simple but frequent example of 1-to-many German–English alignment found in the *Sophie’s World* portion of the SMULTRON corpus (Volk et al., 2010).

Some of the initial research on the subject of projecting word-level annotation from one language to another was published in Yarowsky and Ngai (2001). Here, the authors used IBM Model 3 (Brown et al., 1990) to align large parallel corpora between English–Chinese and English–French. A POS tagger was trained for French using projection from English, and NP bracketers were trained similarly for both French and Chinese. The authors identified noisy statistical alignment and 1-to-many alignments as two main issues in performing projection. The first of these issues is indeed a difficult problem for resource-poor languages, as high-quality statistical word alignment often requires much more bitext than the data available for the language. While it is not a full solution to the problem, many of the language pairs we use in this work are drawn from collection of interlinear glossed text (IGT), as shown in Figure 1, which provides unique shortcuts for obtaining word alignment with a small amount of data. IGT will be discussed further in §2.2.

The second issue of 1-to-many alignments is one that may be the result of linguistic divergence between a language pair where the source is morphologically richer than the target. In cases such as this, finding common patterns of conflation can be useful for generalizing a projection to new data. For instance, Fig. 3 shows a very simple but common case of conflation in the SMULTRON corpus (Volk et al., 2010) where a single German word aligns to multiple English words. Using direct projection alone, the same POS tag would be projected to both English tokens. In this case, using a universal tagset such as those presented by Petrov et al. (2012) could help alleviate the problem, but for more complex cases, learning the pattern would be more critical.

Hwa et al. (2004) investigate the issues in using projection between languages to develop and train syntactic parsers, and outline the Direct Correspondence Assumption (DCA), the assumption made in projection algorithms that the target language tree should be homomorphic to the source language tree. While useful, this assumption often does not hold, as the authors point out. In order to fix some of the errors made by faulty projections, Hwa et al. use an approach that applies post-projection correction rules. For projection from English to Spanish, the accuracy of the projected structures increased from 36.8% to 70.3%. The accuracy of the English to Chinese projection increased from 38.1% and 67.3%.

While these language-specific rewrite rules are promising, they still require language-specific knowledge. What we seek to accomplish in this paper is a general framework for automatically detecting this divergence, both in specific language pairs, and its frequency throughout a large number of languages. With the use of this automated divergence detection, it may be possible to learn these rewrite rules from a small annotated corpus and use them to improve projection algorithms.

2.2 Interlinear Glossed Text

As mentioned in the preceding section, much of the data for our experiments was drawn from the unique IGT data type. IGT instances are a common way for linguists to give illustrative examples for a language being studied. Figure 1 shows an instance of IGT for Hindi and English. Of special interest in IGT instances is the middle gloss line which gives a word-by-word gloss of the source language. This annotated gloss typically contains words from the English translation, albeit reordered and with extra morphological and semantic information. The matching of gloss and translation can be utilized to obtain high-quality automatic word alignment between source and target, without the need for far larger amounts of data typically required by statistical alignment algorithms.

In Xia and Lewis (2007), enriched IGT data for 7 language pairs was created using this augmented alignment and structural projection, then hand-corrected to create gold standards with minimal expert intervention. They showed the potential for using IGT as a resource for languages for which finding resources would otherwise be extremely difficult or impossible to obtain. We will use this data for the current work. A breakdown of the language pairs can be seen in §4.1.

Lewis and Xia (2008) used projected phrase structures to determine the basic word order for 97 languages using a database of IGT instances. By using the alignment method described above and projecting phrase structures from English to the language line, the word order in the foreign language could be inferred. For languages with just 10–39 IGT instances, the accuracy of predicting basic word order was 79%; with more than 40 instances, the accuracy jumped to 99%.

2.3 Linguistic Divergence

These studies illustrate the promise of projection for bootstrapping new tools for resource-poor languages, but one limitation is their reliance on the assumption that syntactic structures of the two sentences in a given sentence pair are similar. While Hwa et al.’s *Direct Correspondence Assumption* (DCA) describes the assumption made for projection, Dorr (1994) makes a deeper analysis of divergence in languages. Dorr outlines *lexical conceptual structures*

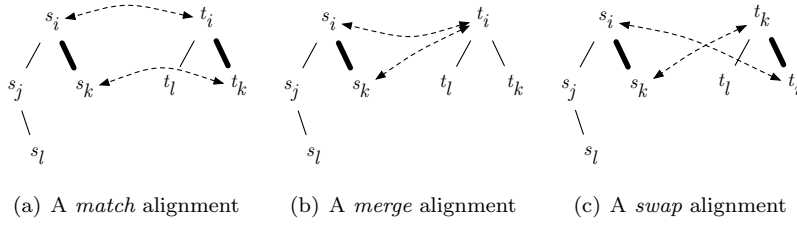


Fig. 4 Definition of a *match*, *merge*, and *swap* edges in a tree pair.

(LCS) that provide a general framework to describe these exceptions to the DCA. This framework is capable of representing divergence stemming from syntactic, lexical, or semantic differences, but for the purposes of this paper we will focus primarily on those that are lexical and syntactic.

Dorr identifies a number of ways in which languages may diverge, specifically syntactic and semantic differences in mappings between languages. Our goal in this work is to create a methodology by which some common types of divergences can be detected automatically from bitexts in order to improve the performance of existing structural projection methods.

3 Methodology

In our approach to automatically detecting divergent structures between language pairs, we first propose a metric to measure the degree of *matched* edges between source and target trees (§3.1). Second, we define three operations on trees in order to capture three common types of divergence (§3.2). Third, we apply the operations on a tree pair and show how the operations could affect the degree of tree *match* (§3.3). After explaining the relationship of our operations to Dorr’s divergence types (§3.4), we discuss how knowing these divergence types can be useful in improving structural projection algorithms (§3.5).

3.1 Comparing Dependency Trees

One of the key aspects of our method was devising a metric to compare dependency trees cross-linguistically, as most existing tree similarity measures are intended to compare tree representations with the same number of tokens. Comparing between languages, on the other hand, means that the number of tokens can vary. We instead look for a method to determine similarity by means of matched edges in the tree, as shown in Figure 4.

Given an IGT, let F be the parse tree for the language line (aka source tree), E be the parse tree for the translation line (aka target tree), and A be the word alignment between the language line and the translation line. F is

made up of the words W , in the language line, and a set of edges between the words, as follows:

$$F = (W_F, T_F) \quad (1)$$

$$W_F = (f_1 \dots f_n) \quad (2)$$

$$T_F = \{(f_i, f_k) \dots (f_n, f_m)\} \quad (3)$$

E is defined similarly, except words in the translation line are denoted as e_i , not f_i . The alignment A is a set of word pairs:

$$A = \{(f_i, e_k) \dots (f_j, e_l)\} \quad (4)$$

We call an (F, E, A) tuple an aligned tree pair. A Corpus, C , in our experiments, is a set of (F, E, A) tuples.

An edge (f_i, f_k) in the foreign-language tree is said to match an edge (e_i, e_k) in the english-language tree if f_i is aligned to e_i and f_k is aligned to e_k . Because the alignment between a sentence pair can be many-to-many, we define the following functions, which map a word from one sentence to the set of words in the other sentence.

$$R_{F \rightarrow E}(f_i, A) = \{e | (f_i, e) \in A\} \quad (5)$$

$$R_{E \rightarrow F}(e_i, A) = \{f | (f, e_i) \in A\} \quad (6)$$

We then define the boolean function *match*, as follows:

$$match(f_i, f_j, T_E, A) = \begin{cases} 1 & \text{if } \exists e_a, e_b \left((e_a \in R_{F \rightarrow E}(f_i)) \wedge \right. \\ & \left. (e_b \in R_{F \rightarrow E}(f_j)) \wedge ((e_a, e_b) \in T_E) \right) \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

That is, an edge (f_i, f_j) in F matches some edge in E according to A if there exists two target words, e_a and e_b in E such that e_a aligns to f_i , e_b aligns to f_j , and (e_a, e_b) is an edge in E .

Given an aligned tree pair (F, E, A) , we define *SentMatch* (F, E, A) as the percentage of edges in F that match some edge in E . Given a corpus C , we define *CorpusMatch* $_{src \rightarrow tgt}(C)$ as the percentage of edges in the source trees that match some edges in the corresponding target trees. Similarly, *CorpusMatch* $_{tgt \rightarrow src}(C)$ is the percentage of edges in the target trees that match some edges in the corresponding source trees.

$$CorpusMatch_{Src \rightarrow Tgt}(C) = \frac{\sum_{(F,E,A) \in C} \left(\sum_{(f_i, f_j) \in T_F} match(f_i, f_j, T_E, A) \right)}{\sum_{(F,E,A) \in C} |T_F|} \quad (8)$$

3.2 Defining Tree Operations

When an edge (f_i, f_k) in a tree does not match any edge in the other tree, it may be caused by one of the following cases:

- C1. f_i or f_k are spontaneous (they do not align with any words in the other tree).
- C2. f_i and f_k are both aligned with the same node e_i in the other tree (Fig 4(b)).
- C3. f_i and f_k are both aligned with nodes in the other tree, e_k and e_i , but in a reversed parent-child relationship (Fig 4(c)).
- C4. There is some other structural differences not caused by C1–C3.

The first three cases are common. To capture them, we define three operations on a tree — *remove*, *merge*, and *swap*.

O1 – Remove

The *remove* operation is used to remove spontaneous words. As shown in Figure 5(a), removal of the node l is accomplished by removing the link between node l and its parent, j , and adding links between the parent and the removed node’s children.

This result of this operation looks can be seen in Figure 5(a), using the relation *Children*, which maps a word to the set of all its children in the tree.

```

1 Algorithm: Remove( $w, T$ )
   Input  :  $T = \{(w_i, w_j) \dots (w_m, w_n)\}$  ;           // Input tree
   Input  :  $w$  ;                                           // Word to remove.
   Output:  $T'$  ;                                           // Modified tree
2 begin
3    $T' = T - \{(w, w_i) | w_i = \text{parent}(w, T)\}$            // Remove edge between  $w$  and parent  $w_i$ 
4    $- \{(w_j, w) | w = \text{parent}(w_j, T)\}$                  // Remove edges for children of  $w$ 
5    $+ \{(w_j, w_i) | w_i = \text{parent}(w, T), w = \text{parent}(w_j, T)\}$  ;
   /* Finish by ‘promoting’ former children of  $w$  to now attach to  $w$ ’s parent,
       $w_i$ . */
6 return  $T'$ 

```

Algorithm 1: Remove a token w from the tree T .

O2 – Merge

The *merge* operation is used when a node and some or all of its children in one tree align to the same node(s) in the other tree, as can be seen in Figure 4(b). The parent j and child l are collapsed into a merged node, as indicated by $l+j$ in Figure 5(b), and the children of l are promoted to become children of the new node $l+j$. The result can be seen in Figure 5(b).

```

1 Algorithm: Merge( $w_c, w_p, T$ )
   Input  :  $T = \{(w_i, w_j) \dots (w_m, w_n)\}$  ; // Input tree
   Input  :  $w_c$  ; // Child word to merge.
   Input  :  $w_p$  ; // Parent word to merge.
   Output:  $T'$  ; // Modified tree
2 begin
3    $T' = T - \{(w_c, w_p)\}$ 
4    $- \{(w_i, w_c) | w_c = \text{parent}(w_i, T)\}$ 
5    $+ \{(w_i, w_p) | w_c = \text{parent}(w_i, T)\}$ ;
6 return  $T'$ 

```

Algorithm 2: Merge a child w_c and parent w_p in the tree T , and “promote” the children of w_c to be children of w_p .

O3 – Swap

The *swap* operation is used when two nodes in one tree are aligned to two nodes in the other tree, but in a reciprocal relationship, as shown in Figure 4(c). This operation can be used to handle certain divergence types such as *demotional* and *promotional* divergence, which will be discussed in more detail in §3.4.

Figure 5(c) illustrates how the swap operation takes place by swapping nodes l and j . Node j , the former parent, is *demoted*, keeping its attachment to its children. Node l , the former child, is *promoted*, and its children become siblings of node j , the result of which can be seen in Figure 5(c).

```

1 Algorithm: Swap( $w_c, w_p, T$ )
   Input  :  $T_L = \{(w_i, w_j) \dots (w_m, w_n)\}$  ; // Input tree
   Input  :  $w_c$  ; // Child word to swap.
   Input  :  $w_p$  ; // Parent word to swap.
   Output:  $T'$  ; // Modified tree
2 begin
3    $T' = T - \{(w_c, w_p)\} + \{(w_p, w_c)\}$  // Swap the order in the  $(w_c, w_p)$  edge
4    $- \{(w_p, w_i) | w_i = \text{parent}(w_p, T)\}$  // Remove edges for parent  $w_p$ 
5    $+ \{(w_c, w_i) | w_i = \text{parent}(w_p, T)\}$  ; // Add edges from  $w_p$ 's parent to  $w_c$ 
6 return  $T'_L$ 

```

Algorithm 3: Swap a child w_c and parent w_p in the tree T .

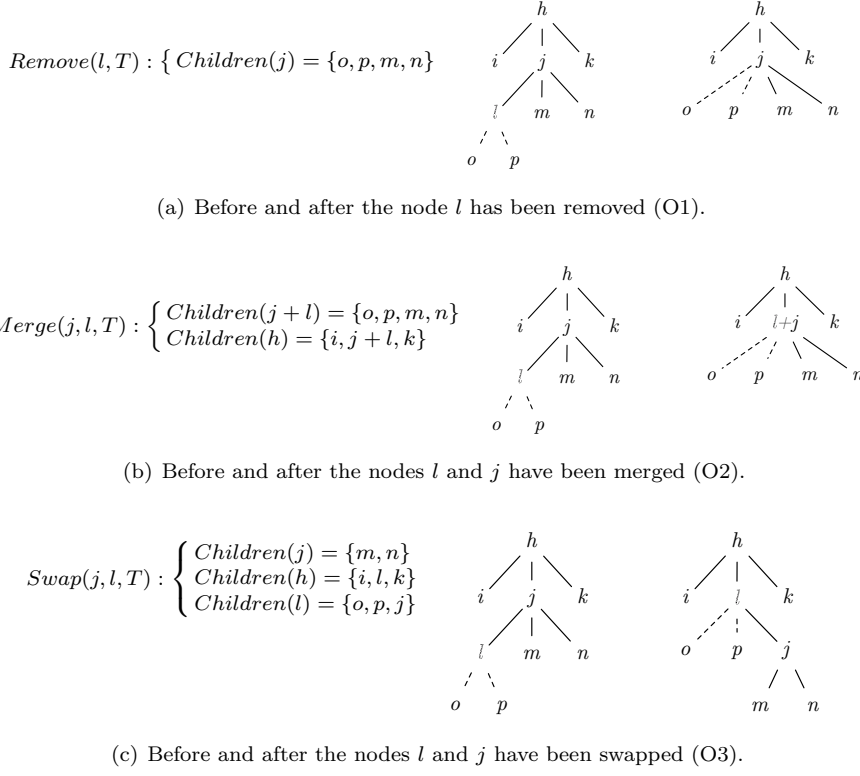


Fig. 5 Trees showing the results of the operations defined in O1–O3. $Children(w)$ returns the set of words that depend on w . Here we show the value of $Children(node)$ after the operations only if its value is changed by the operations.

3.3 Calculating Tree Matches After Applying Operations

The operations O1–O3 are proposed to handle common divergence cases in C1–C3. To measure how common C1–C3 is in a language pair, we design an algorithm that transforms a tree pair based on a word alignment.

The algorithm takes a tree pair (F, E) and a word alignment A as input and creates a modified tree pair (S', T') and an updated word alignment A' as output. It has several steps. First, spontaneous nodes (nodes that do not align to any node on the other tree) are removed from each tree. Next, if a node and its parent align to the same node on the other tree, they are merged and the word alignment is changed accordingly. Finally, the swap operation is applied to a node f_i and its parent f_p in one tree if they align to e_i and e_p respectively and e_p is a child of e_i in the other tree. The pseudocode of the algorithm is shown in Algorithm 4.

Now given a corpus C and word alignment between each sentence pair, we can measure the impact of C1–C3 by comparing $CorpusMatch_{Src \rightarrow Tgt}(C)$

Algorithm 4: Algorithm for altering an aligned tree pair.

```

input :  $c = (F, E, A)$  ; // A parallel sentence with alignment
output:  $c' = (F', E', A')$  ; // modified output sentence.
1 Let  $F = (W_F, T_F)$  ;
2 Let  $E = (W_E, T_E)$  ;
3 Let  $A = \{(f_i, e_j), \dots, (f_k, e_l)\}$  ;
4 begin
    // Step 1(a): Remove spontaneous nodes from  $F$ 
5   foreach  $f_i \in W_F$  do
6     if  $\nexists e_j : (f_i, e_j) \in A$  then
7        $T_F = \text{Remove}(f_i, T_F)$  ; // See Algorithm 1
    // Step 1(b): Remove spontaneous nodes from  $E$ 
8   foreach  $e_j \in W_E$  do
9     if  $\nexists f_i : (f_i, e_j) \in A$  then
10       $T_E = \text{Remove}(e_j, T_E)$  ; // See Algorithm 1
    // Step 2(a): Find nodes to merge in  $F$  and merge them
11   foreach  $(f_i, e_j) \in A$  do
12     Let  $f_p = \text{parent}(f_i, T_F)$  ;
13     if  $(f_p, e_j) \in A$  then
14        $T_F = \text{Merge}(f_i, f_p, T_F)$  ; // See Algorithm 2
15        $A = A - \{(f_i, e_j)\}$  ;
    // Step 2(b): Find nodes to merge in  $E$  and merge them
16   foreach  $(f_i, e_j) \in A$  do
17     Let  $e_p = \text{parent}(e_j, T_E)$  ;
18     if  $(f_i, e_p) \in A$  then
19        $T_E = \text{Merge}(e_j, e_p, T_E)$  ; // See Algorithm 2
20        $A = A - \{(f_i, e_j)\}$  ;
    // Step 3: Find nodes to swap in  $F$  and swap them
21   foreach  $(f_i, e_j) \in A$  do
22     Let  $f_p = \text{parent}(f_i, T_F)$  ;
23     if  $\exists e_c : e_j = \text{parent}(e_c, T_E)$  and  $(f_p, e_c) \in A$  then
24        $T_F = \text{Swap}(f_i, f_p, T_F)$  ; // See Algorithm 3
25   return  $(F', E', A')$  ;

```

scores before and after applying operations O1–O3. This process can also reveal some patterns of divergence (e.g., what types of nodes are often merged), and the patterns can later be used to enhance existing projection algorithms.

3.4 Relationship to Dorr (1994)

Dorr (1994) lists seven types of divergence for language pairs. While our analysis method is more coarse-grained than the Lexical Conceptual Structure (LCS) that Dorr proposes, it is nonetheless able to capture some of the same cases.

For instance, Figure 6 illustrates an example of what Dorr identified as “promotional” divergence, where *usually*, a dependent of the verb *goes* in En-

English, is “promoted” to become the main verb, *suele* in Spanish. In this case, the direction of the dependency between *usually* and *goes* is reversed in Spanish, and thus the *swap* operation can be applied to the English tree and result in a tree that looks very much like the Spanish tree. A similar operation is performed for *demotional* divergence cases, such as aligning “I like eating” with the German translation “*Ich esse gern*” (“I eat likingly”). Here, the main verb in English (“like”) is *demoted* to an adverbial modifier in German (“*gern*”). The *swap* operation is applicable to both types of divergence and treats them equivalently, and so it essentially can handle a superset of promotional and demotional divergence, namely, “head-swapping.”

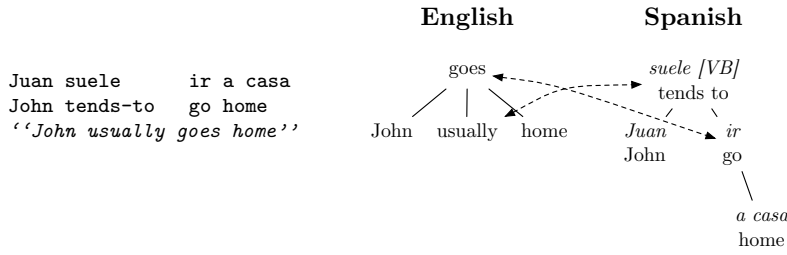


Fig. 6 An example of *promotional* divergence from Dorr (1994). The reverse in parent-child relation is handled by the *Swap* operation.

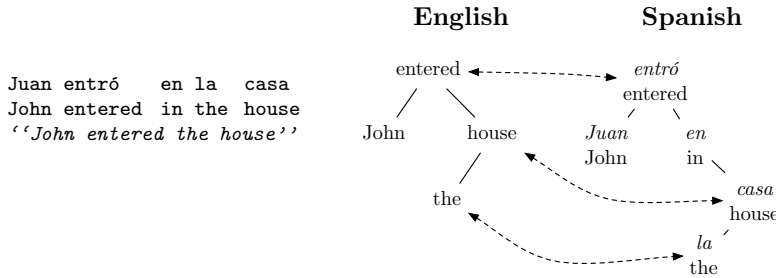


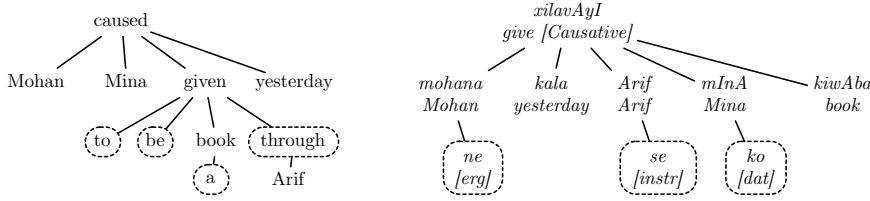
Fig. 7 Example of structural divergence, which is handled by the *remove* operation.

Another type of divergence that can be captured by our approach is Dorr’s “structural” divergence type, as illustrated in Figure 7. The difference between the English and Spanish structures in this case is the form of the argument that the verb takes. In English, it is a noun phrase; in Spanish, it is a prepositional phrase. While the tree operations defined previously do not explicitly recognize this difference in syntactic labels, the divergence can be handled by the *remove* operation, where the spontaneous “*en*” in the Spanish side is removed.

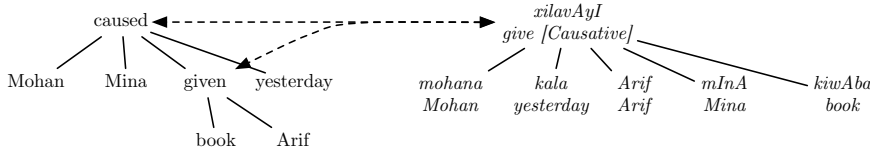
Next, Dorr’s description of *conflational* divergence lines up well with the *merge* operation (see Fig 5(b)). Figure 8 illustrates an example for English and Hindi, where both sides have spontaneous words (e.g., *to* and *a* in English) and a causative verb in Hindi corresponds to multiple verbs in English. Figure 8(b) shows the original tree pair, 8(c) demonstrates the altered tree pair after removing spontaneous words from both sides. Figure 8(d) shows the tree pairs after the English verbs are merged into a single node. It is clear that the *remove* and *merge* operations make the Hindi and English trees much similar to each other.

mohana ne kala Arif se mInA ko kiwAba xilavAyI
 Mohan [erg] yesterday Arif [inst] Mina [dat] book give-caus
 ‘‘Mohan caused Mina to be given a book through Arif yesterday.’’

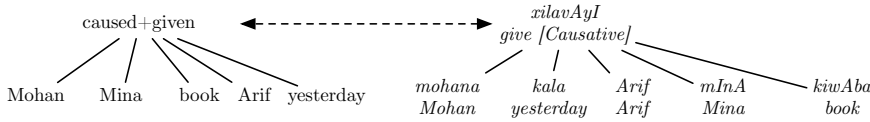
(a) Interlinear text of a sentence pair.



(b) Initial trees showing spontaneous words on both sides.



(c) Altered trees after removing spontaneous words from both sides, and showing conflational divergence between multiple English words and a single Hindi word.



(d) Altered trees after merging multiple words on the English side.

Fig. 8 Case of conflational divergence, handled by remove and merge operations.

In addition to the four divergence types mentioned above, additional operations could be added to handle other divergence types. For instance, if dependency types (e.g., patient, agent) are given in the dependency structure, we can define a new operation that changes the dependency type of an edge

Corpus	Language	# Instances	# Words (F / E)
Hindi Treebank	Hindi	147	963 / 945
ODIN	German	105	747 / 774
	Irish	46	252 / 278
	Hausa	77	424 / 520
	Korean	103	518 / 731
	Malagasy	87	489 / 646
	Welsh	53	312 / 329
	Yaqui	68	350 / 544
SMULTRON	German	281	6829 / 7236
	Swedish	281	8402 / 9377

Table 1 Data set sizes for all languages. For the number of words, the number of words in the foreign (F) language are given first, followed by the number of English (E) words.

to account for *thematic* divergence, where thematic roles are switched as in “I like Mary” in English vs. “*María me gusta a mí*” (Mary pleases me) in Spanish. Similarly, an operation that changes the POS tag of a word can be added to cover *categorical* divergence where words representing the same semantic content have different word categories in the two languages, such as in “I am hungry” in English versus “*Ich habe Hunger*” (I have hunger) in German.

Compared to Dorr’s divergence types, whose identification requires knowledge about the language pairs, our operations on the dependency structure relies on word alignment and tree pairs and can be applied automatically.

3.5 Extending Projection Algorithms

Using the techniques described above, and following Hwa et al. (2004), we can find post-processing rules automatically. Our method couples the divergent cases C1–C3 with corresponding operations O1–O3. As the operations are applied, statistics are kept on the nodes that are affected, and thus common divergence patterns can be detected by analyzing this data. By generalizing the data found in this analysis, rules that can handle common divergence types could be applied to particular language pairs that exhibit such patterns in the small training corpus. We will discuss one such attempt at learning a rewrite rule in §4.4.

4 Experiments

With the matching function and tree operations defined in the previous section, we looked at a total of eleven language pairs, using the corpora described in Table 1.

4.1 Data

Our work utilizes three corpora for a total of eleven language pairs. The corpora used are the SMULTRON treebank (Volk et al., 2010), the guideline sentences in IGT form from the Hindi treebank (Bhatt et al., 2009), and several sets of IGT data as used in (Xia and Lewis, 2007). The statistics of the corpora are shown in Table 1. Ten of the language pairs use English as one side of the language, while the eleventh uses the pair of German and Swedish from the SMULTRON corpus.

In the SMULTRON Treebank, the German and Swedish phrase trees are marked for head children, allowing for the automatic extraction of dependency trees. The English side of the phrase structures do not contain edge labels and are instead converted to dependency trees using a head percolation table (Collins, 1999).

From the Hindi Treebank guidelines, we extracted example sentences in the form of IGT (i.e., Hindi sentences, English gloss, and English translation) and the Hindi dependency structures manually created by the guideline designers. We obtained dependency structures for the English translation by running the Stanford dependency parser de Marneffe et al. (2006) and then we hand corrected the structures. Word alignment is initially derived from the IGT instances using heuristic alignment following Xia and Lewis (2007), and later hand-corrected. The IGT data from Xia and Lewis (2007) was obtained in the manually corrected dependency forms described in §2.2.

4.2 Match Results

By running Algorithm 5, we can calculate the $CorpusMatch_{Src \rightarrow Tgt}$ and $CorpusMatch_{Tgt \rightarrow Src}$ before and after each operation and see how the operation affects the percentage of matched edges in the corpus. As the operations are applied, the percentage of matches between the trees should increase until all the divergence cases that can be handled by operations O1–O3 have been resolved. At this point, the final match percentage can be seen as an estimate of the upper-bound on performance of a simple projection algorithm, if C1–C3 can be identified and handled by O1–O3. Table 2 shows the full results of this process for English and Hindi, while Table 3 shows a summary for the results in the remaining ten languages.

4.3 Operation Breakdown By POS

After performing the operations as seen in §4.2, we can get further insight into what precisely is happening within each language by breaking down the operations by the POS tags on which the operations apply. Table 4 shows some of these POS tag breakdowns for a number of languages, and the frequency with which the given operation applies to the POS tag or POS tag pair out of

English \leftrightarrow Hindi

	English \rightarrow Hindi				
	Match	Swap	Unaligned	Merge	Edges
Baseline	47.7	9.1	20.9	1.6	794
Remove	66.1	11.7	0.0	2.1	622
Merge	69.5	12.3	0.0	0.0	586
Swap	90.3	0.3	0.0	0.0	586
	Hindi \rightarrow English				
	Match	Swap	Unaligned	Merge	Edges
Baseline	46.3	5.6	20.7	1.7	816
Remove	63.4	5.3	0.0	2.2	647
Merge	69.2	4.6	0.0	0.0	587
Swap	89.9	2.4	0.0	0.0	587

Table 2 Breakdown of edges as operations are applied to the English \leftrightarrow Hindi language pair. The “Edges” column represents the number of total edges in the trees of the left hand of the language pair. The numbers given in the other columns are the percentages of those edges that are either in a *match*, *swap*, or *merge* alignment, or the edges for which the child is *unaligned*.

	ODIN DATA							SMULTRON DATA		
	DE	GD	HA	MG	KO	CY	YAQ	DE	SV	DE-SV
Baseline	76.7	72.0	54.4	57.4	56.0	75.4	54.4	40.7	37.5	43.3
Remove	93.9	87.8	95.7	88.9	88.1	95.1	90.9	63.6	62.2	73.5
Merge	95.4	92.5	97.5	97.4	95.4	97.2	95.9	71.8	73.9	82.8
Swap	96.8	94.1	97.5	98.0	96.1	98.2	96.2	83.0	84.2	87.2

Table 3 Summary of the results for the remaining ten language pairs, German (DE), Scots Gaelic (GD), Hausa (HA), Malagasy (MG), Korean (KO), Welsh (CY), Yaqui (YAQ), Swedish (SV) and DE-SV. Except for DE-SV, English is the first language of the pair.

all the times it is seen in that language. Table 4 shows phenomena from the language pairs that we would hope to see. For instance, rows 6 and 7 show the English \rightarrow German pair merging many nouns as multiple English words are expressed as compounds in German. In another case, Row 8 shows that all Hindi Nouns undergo *swap* with prepositions, as Hindi uses postpositions. Noticing this extremely high regularity leads us to the next experiment, where we examine how such regularly-occurring rules might be harnessed to improve projection.

4.4 Analyzing Trees For Post-Processing Rules

Given that we can break down the operations as they apply to particular projected POS tag sequences, we also would like to see if the detected patterns can be used after projection, to correct the projected trees. In Table 5, 80% of the Hindi data was used to train a small corpus on swap patterns were detected. The projection algorithm was run on the remaining 20%. The first row of Table 6 shows the results of the baseline projection algorithm.

Next, we used the dependencies as projected into Hindi from the English trees as shown in Table 5 and automatically selected those that occurred with 80% or higher frequency. The projection algorithm was run as usual, and then

Algorithm 5: Calculating the percentage of matched edges in a corpus C .

```

input : A corpus  $C$ 
output:  $CorpusMatch_{Src \rightarrow Tgt}(C)$ 
          $CorpusMatch_{Tgt \rightarrow Src}(C)$ 

1 begin
2   Let  $F \rightarrow E$  matches = 0 ;
3   Let  $E \rightarrow F$  matches = 0 ;
4   foreach  $(F, E, A) \in C$  do
5     Let  $F = (W_F, T_F)$  ;
6     Let  $E = (W_E, T_E)$  ;
7     Let  $A = \{(f_i, e_j), \dots, (f_k, e_l)\}$  ;
      // Get matches for  $F \rightarrow E$ 
8     foreach  $(f_c, f_p) \in T_F$  do
9       /* If the child and parent in this edge align with the child→parent
10        edge of the other tree... */
11       if  $\exists e_c, e_p : e_p = \text{parent}(e_c, T_E)$ 
12         and  $(f_p, e_p) \in A$ 
13         and  $(f_c, e_c) \in A$ 
14       then
15         // Increase the match count.
16          $(F \rightarrow E \text{ matches})++$ ;
      // Get matches for  $E \rightarrow F$ 
17     foreach  $(e_c, e_p) \in T_E$  do
18       /* If the child and parent in this edge align with the child→parent
19        edge of the other tree... */
20       if  $\exists f_c, f_p : f_p = \text{parent}(f_c, T_F)$ 
21         and  $(f_p, e_p) \in A$ 
22         and  $(f_c, e_c) \in A$ 
23       then
24         // Increase the match count.
25          $(E \rightarrow F \text{ matches})++$ ;
26   return  $Match(F \rightarrow E) = 100 \times \frac{(F \rightarrow E \text{ matches})}{|T_F|}$  ;
27   return  $Match(E \rightarrow F) = 100 \times \frac{(E \rightarrow F \text{ matches})}{|T_E|}$  ;

```

every dependency that matched the given pattern was swapped to correct the trees. The second row of the table shows that applying all rules above this threshold actually brought the F_1 score down from 64.95 to 63.59. To ensure that only swaps which were both regular and frequent occurred, we tightened the restriction so that only rules that occurred three or more times were chosen. This resulted in an improvement to an F_1 score of 73.35 as shown in the final row of Table 6.

4.5 Remaining Cases

After applying three operations, there may still be unmatched edges. An example is given in Figure 9.¹ The dependency edge $(in, America)$ can be reversed by the *swap* operation to match the Hindi counterpart. The difficult part is the

¹ It is a topic of debate whether *mentally* in English should depend on *in* or *am*. If it depends on *in*, handling the divergence would be more difficult.

Merges				
Lang Pair	Row #	Child POS	Parent POS	% Merge
Eng→Hin	1	MD	VB	42.9%
	2	NN	NN	14.3%
Hin→Eng	3	VAUX	VM	45.4%
	4	NN	VM	5.5%
Eng→Ger	5	NN	NNS	66.7%
	6	NN	NN	65.4%
	7	NNS	NN	0%
Swaps				
Hin→Eng	8	NN	IN	100%
	9	NNP	IN	20.0%
Ger→Eng	10	NN	APPRART	72.7%
	11	CC	NN	61.5%
Removals				
Eng→Hin	12	DT		86.4%
	13	TO		75.6%
Hin→Eng	14	PSP		69.8%
	15	VAUX		18.6%
Eng→Ger	16	POS		57.1%
	17	DT		20.2%
Ger→Eng	18	PRF		85.2%
	19	ADV		43.9%

Table 4 Breakdown of significant merge and swap statistics for various language pairs, where the language to the left of the arrow is the one being altered.

Projected Dependency	Swap %	# Dependencies
NN→IN	100%	33
NNP→IN	100%	12
JJ→VBD	100%	1
VBZ→VBG	60%	5
VBG→VBZ	28.57%	7
VBG→NN	5%	20

Table 5 Sampling of dependency links which were projected from English onto Hindi. The “Swap %” measures the number of times that particular dependency was found in a swapped alignment with the English tree.

Projection Accuracy for English→Hindi				
Threshold %	Count	Precision	Recall	F ₁
N/A	N/A	65.64	64.28	64.95
80%	0	64.26	62.93	63.59
80%	3	74.13	72.59	73.35

Table 6 Results of automatically detecting and applying projected swaps between English and Hindi. The threshold % describes the rate at which a rule must apply to the POS tag pair before it is automatically applied, and the Count column describes how many times the rule must occur before it is considered. The first row is the baseline with no rules used.

adverb *mentally* in English corresponds to the noun *mana* (*mind*) in Hindi. If the word alignment includes the three word pairs as indicated by the dotted lines, one potential way to handle this kind of divergence is to extend the definition of *merge* to allow edges to be merged on both sides simultaneously – in this case, merging *am* and *mentally* in the English side, and *hE* (*is*) and *mana* (*mind*) on the Hindi side.

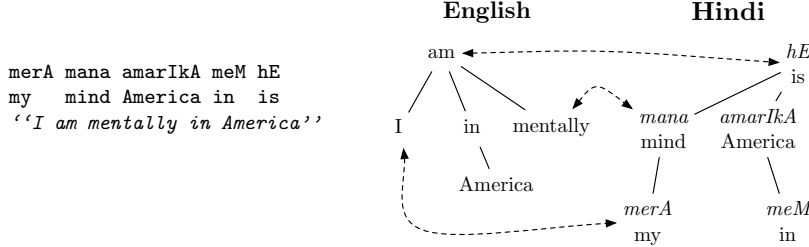


Fig. 9 A tree pair that still has unmatched edges after applying the algorithm in Table 4. The dotted line indicates word alignment that would be needed to resolve the divergence with the *extended* merge operation.

5 Discussion of Results

The results of the experiments above show that the match scoring that we have introduced here has the potential to address many interesting issues arising between language pairs. In this section, we highlight some interesting observations based on the experimental results.

5.1 Match Scores

The results of tables 2 and 3 are interesting in comparing similarity both across languages and corpora. For instance, in the scores for the baseline ODIN data, we see that the baseline for matches between English and German is the highest out of all the pairs at 76.7%. Scots Gaelic and Welsh are 72% and 75.4%, respectively. Hausa, Malagasy, Korean, and Yaqui all show baseline scores between 54–57%. This seems in line with what we would expect, with German and the Celtic languages being closely related to English, and the others being quite unrelated.

Another stark contrast can be seen between all the languages in the ODIN data and the languages in the SMULTRON corpus. While the ODIN sentences tend to be short sentences used primarily for illustrative purposes, the SMULTRON corpus consists of economic, literary, and weather domains. As Table 1 shows, the SMULTRON sentences are much longer on average. A closer look at the SMULTRON translations also shows them to be much freer translations

than those found in the ODIN data. While the size of the data sets used here are very small, and the ODIN IGT data may be biased towards illustrative purposes (described as the “IGT Bias” in Xia and Lewis (2007)), it would appear that these results illustrate that the match detection is capable of two interesting corpus analyses. First, by comparing baselines match results among comparable corpora, basic similarities between languages appear to pattern as expected. Second, the freer translations in the SMULTRON data appear with lower scores all across.

One final item of interest from the match results can be seen in the Hindi data in Table 2. Here, there appears to be a large jump after the *swap* operation has been performed. Seeing as swapping such as this would be problematic for projection algorithms, this is the inspiration for automatically inferring the swap rules in §4.4.

5.2 POS Breakdowns

The breakdown of the operations by language and POS in Table 4 provides a good opportunity to check that the defined operations conform with expectations for specific languages.

For instance, Row 1 in Table 4 shows Modals (MD) merging with a parent (VB). This is in line with instances such as Figure 8(c) where Hindi combines aspect with a verb that is typically expressed as a separate word in English. This does not appear to be a very frequent occurrence, however, as it only occurs for 42.9% of MD→VB dependencies.

Row 3, going from Hindi to English shows the case where auxiliary verbs VAUX merge with main verbs VM. These cases typically represent those where Hindi represents tense as an auxiliary verb, whereas English tense is expressed by inflection on the verb.

With regard to spontaneous words in English and Hindi, Row 14 shows that 69.8% of case markers (PSP) were removed from Hindi that were either absent in English or applied as inflections to the noun, while 86% of determiners in English were removed, as they are not seen in Hindi (Row 12).

Examining the English and German data in Table 4, we first see in Row 5 that 66.7% of NN-NNS dependencies in English merge. This, along with the 65.4% of NN-NN dependencies merging, is something we would expect to see in German, as it compounds nouns with far more frequency than English. Interestingly, as row 7 shows, a plural noun child never merges with a parent noun.

Finally, looking more closely at the swaps, we see a surprising 100% of NN→IN dependencies are swapped in Hindi, giving further impetus for the rules as described in §4.4.

5.3 Automatically Created Rules

As Table 6 illustrates, the potential for automatically created post-processing rules for a projection algorithm is clearly shown by the 13% increase in F_1 score from the uncorrected projection algorithm. While this example focuses only on the easily correctable swap operation, the success here suggests that a similar analysis of the merge operations could be used to correct the 1-to-many projections from English onto other languages that Yarowsky and Ngai (2001) cited as problematic.

5.4 Discussion of Issues

Two large issues that our methodology faces are data sparsity and translation quality of the sentence pairs in the data sets. The former is somewhat inevitable given the task—a reasonable amount of annotated data is not always likely to exist for languages with scarce electronic resources, and guaranteeing coverage is difficult. As with the Hindi data, however, using IGT as a resource has convenience in both covering wide varieties of phenomena in a language, and providing a gloss that assists in creating word-level alignments. Creating dependency annotation on a small set of data from a source like ODIN (Lewis, 2006) can get a lot of mileage with a small amount of investment.

Perhaps the more challenging issue is determining whether divergence in a language pair is caused by fundamental differences between the languages, or simply stylistic choices in translation. The latter of these scenarios appeared to be common in portions of the SMULTRON data, where translations appeared to be geared toward naturalness in the target language; in contrast, the translations in the Hindi guideline sentences were intended to be as literal as possible. Again, IGT provides a good possible solution, as such examples are often intended specifically for illustrative purposes.

6 Conclusion and Future Work

In this paper, we have demonstrated a generalizable approach to detecting patterns of structural divergence across language pairs using simple tree operations based on word alignment. We have shown that this methodology can be used to detect similarities between languages on a course level, as well as serve as a general measure of similarity between dependency corpora. Finally, we showed that harnessing these detection methods has potential for improving projection algorithms with little to no expert involvement.

This work further shows that there is still plenty of room for improvement in existing projection methods. In future work, we plan to investigate further ways of learning post-processing rules in projected trees, such as how to properly correct for 1-to-many alignments, and methods for reattaching spontaneous words in the foreign language. In future work, we would also like to

examine how labeled dependency edges may play a role in describing divergences between languages, and how these might be more adequately corrected for.

Finally, while Georgi et al. (2012) demonstrated that projected trees can be improved upon by using the projected edges as features in a statistical parser, we would like to follow up on this work by examining how the alignment types outlined in this paper might be also used as features in a parser to further improve dependency parsing for resource-poor languages.

The techniques described here are promising for maximizing the effectiveness of existing resources such as IGT for languages with little more available. While the amount of electronic resources continues to increase for those languages in which electronic communication is increasingly common, many of these resource-poor languages are still left behind. Though projection techniques may not ultimately be full replacements for large treebank projects, the ability of these techniques to be rapidly deployed is extremely useful for researchers seeking to experiment with new languages at minimal cost.

References

- Yassine Benajiba and Imed Zitouni. Enhancing mention detection using projection via aligned corpora. In *2010 Conference on Empirical Methods in Natural Language Processing*, pages 993–1001. Association for Computational Linguistics, 2010.
- Rajesh Bhatt, Bhuvana Narasimhan, Martha Palmer, Owen Rambow, Dipti Misra Sharma, and Fei Xia. A multi-representational and multi-layered treebank for Hindi/Urdu. In *The Third Linguistic Annotation Workshop (The LAW III) in conjunction with ACL/IJCNLP 2009*. Association for Computational Linguistics, August 2009.
- Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Fredrick Jelinek, John D Lafferty, Robert L Mercer, and Paul S Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, June 1990.
- Nicoletta Calzolari, Riccardo Del Gratta, Gil Francopoulo, Joseph Mariani, Francesco Rubino, Irene Russo, and Claudia Soria. The LRE Map. Harmonising Community Descriptions of Resources. In *LREC (International Conference on Language Resources and Evaluation)*, Istanbul, 2012.
- Michael Collins. *Head-driven statistical models for natural language parsing*. PhD thesis, University of Pennsylvania, 1999.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC 2006*, 2006.
- Bonnie Jean Dorr. Machine translation divergences: a formal description and proposed solution. *Computational Linguistics*, 20:597–633, December 1994.
- Ryan Georgi, Fei Xia, and William D Lewis. Improving Dependency Parsing with Interlinear Glossed Text and Syntactic Projection. In *Proceedings of*

- the 24th International Conference on Computational Linguistics (COLING 2012)*, Mumbai, India, December 2012.
- Rebecca Hwa, Philip Resnik, Amy Weinberg, and Okan Kolak. Evaluating translational correspondence using annotation projection. In *Proceedings of ACL 2002*, July 2002.
- Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 1(1):1–15, 2004.
- William D Lewis. ODIN: A model for adapting and enriching legacy infrastructure. In *e-Science '06*, page 137, 2006.
- William D Lewis and Fei Xia. Automatically identifying computationally relevant typological features. In *Proceedings of IJCNLP 2008*, 2008.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. A universal part-of-speech tagset. In *Proceedings of LREC*, May 2012.
- Martin Volk, Anne Göhring, Torsten Marek, and Yvonne Samuelsson. SMULTRON (version 3.0) — The Stockholm MULtilingual parallel TReebank, 2010. URL http://www.cl.uzh.ch/research/paralleltreebanks/smultron_en.html. An English-French-German-Spanish-Swedish parallel treebank with sub-sentential alignments.
- Fei Xia and William D Lewis. Multilingual structural projection across inter-linear text. In *Human Language Technologies: NAACL 2007*, 2007.
- David Yarowsky and Grace Ngai. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Proceedings of NAACL*, Stroudsburg, PA, 2001. Johns Hopkins University.